Minimizing Slowdown in Heterogeneous Size-Aware Dispatching Systems (full version)[†]

Esa Hyytiä, Samuli Aalto and Aleksi Penttinen
Department of Communications and Networking
Aalto University School of Electrical Engineering, Finland

***Abstract*—We consider a system of parallel queues where tasks are assigned (dispatched) to one of the available servers upon arrival. The dispatching decision is based on the full state information, i.e., on the sizes of the new and existing jobs. We are interested in minimizing the so-called mean slowdown criterion corresponding to the mean of the sojourn time divided by the processing time. Assuming no new jobs arrive, the shortest-processing-time-product (SPTP) schedule is known to minimize the slowdown of the existing jobs. The main contribution of this paper is three-fold: 1) To show the optimality of SPTP with respect to slowdown in a single server queue *under Poisson arrivals*; 2) to derive the so-called size-aware value functions for M/G/1-FIFO/LIFO/SPTP/SPT/SRPT *with general holding costs* of which the slowdown criterion is a special case; and 3) to utilize the value functions to derive efficient dispatching policies so as to minimize the mean slowdown in a heterogeneous server system. The derived policies offer a significantly better performance than e.g., the size-aware-task-assignment with equal load (SITA-E) and least-work-left (LWL) policies.**

I. Introduction

Dispatching problems arise in many contexts such as manufacturing sites, web server farms, super computing systems, and other parallel server systems. In a dispatching system jobs are assigned upon arrival to one of the several queues as illustrated in Fig. 1. Such systems involve two decisions: (i) *dispatching policy* α chooses the server, and (ii) *scheduling discipline* the order in which the jobs are served. The dispatching decisions are irrevocable, i.e., it is not possible to move a job to another queue afterwards. In the literature, the dispatching problems and their solutions differ with respect to (i) optimization objective, (ii) available information, and (iii) scheduling discipline used in the servers.

Although the literature has generally addressed the mean sojourn time (i.e., response time) as the optimization objective, also other performance metrics can be relevant. One such metric is the *slowdown*[1] of a job, defined as the ratio of the sojourn time and the processing time (service requirement) [17], [9], [1]. The slowdown criterion combines *efficiency* and *fairness* and stems from the idea that *longer jobs can tolerate longer sojourn time*. The optimal scheduling discipline in this respect for a single server queue and a fixed number of jobs is the so-called *shortest-processing-time-product*[2] (SPTP)

[†] This is the full version (including the appendix) of the paper with the same title that appears in the ACM SIGMETRICS 2012, London, UK.

[1]Yang and de Veciana refer to the slowdown as the bit-transmission delay (BTD) [30].

[2]Wierman et al. refer to SPTP as the RS policy, a product of the remaining size and the original size [28].

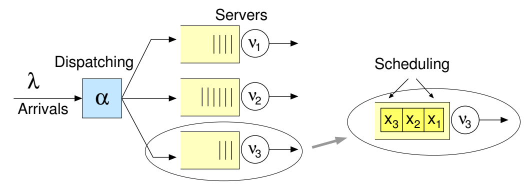


Fig. 1. A dispatching system with $m = 3$ servers.

discipline [30], where the index of a job is the product of the initial and remaining service requirements and the job with the smallest index is served first. With aid of Gittins index, we show that SPTP minimizes the mean slowdown in single server queues also in the dynamic case of Poisson arrivals, i.e., it is the optimal discipline for an M/G/1 queue with respect to the slowdown.

Then we derive value functions with respect to *arbitrary job specific holding costs* for an M/G/1 with the first-in-first-out (FIFO), the last-in-first-out (LIFO), the shortest-processing-time (SPT), the shortest-remaining-processing-time (SRPT) and SPTP scheduling disciplines, where the last three are size-aware. A value function essentially characterizes the queue state in terms of the expected costs in infinite time horizon for a fixed scheduling discipline. In this respect, our work generalizes the results of [19] to arbitrary job specific holding cost rates, and includes also the analysis of the slowdown specific SPTP scheduling discipline.

Finally, we apply the derived value functions to the dispatching problem so as to minimize the mean slowdown when the scheduling disciplines in each queue are fixed. We assume that the dispatcher is fully aware of the state of the system, i.e., of the tasks in each queue and their remaining service requirements. By starting with an arbitrary state-independent policy, we carry out the first-policy-iteration (FPI) step of the Markov decision processes (MDP) and obtain efficient state-dependent dispatching policies.

The rest of the paper is organized as follows. In Section II, we consider a single M/G/1 queue with respect to slowdown criterion, and prove the optimality of SPTP. In Section III, we derive the size-aware value functions with respect to arbitrary job specific holding cost rates for FIFO, LIFO and SPTP (SPT and SRPT are given in the Appendix). The single queue scheduling related results are utilized in Section IV to derive efficient dispatching policies, which are then evaluated in Section V. Section VI concludes the paper.

A. Related Work

FIFO is perhaps the most common scheduling discipline due to its nature and ease of implementation. Other common disciplines are LIFO, SPT and SRPT. As for the objective, minimization of the mean sojourn time has been a popular choice. Indeed, SPT and SRPT, respectively, are the optimal non-preemptive and preemptive schedules with this respect [25]. For a recent survey on fairness and other scheduling objectives in a single server queue we refer to [27].

In the context of dispatching problems, FIFO has been studied extensively in the literature since the early work by Winston [29], Ephremides et al. [8], and others. Often the number of tasks per server is assumed to be known, cf., e.g., *join-the-shortest-queue* (JSQ) dispatching policy [29]. Even though FIFO queues have received the most of the attention, also other scheduling disciplines have been studied. For example, Gupta et. al consider JSQ with *processor-sharing* (PS) scheduling discipline in [12].

Only a few optimality results are known for the dispatching problems. Assuming exponentially distributed interarrival times and job sizes, [29] shows that JSQ with FIFO minimizes the mean waiting time when the number in each queue is available. Also [8] argues for the optimality of JSQ/FIFO when the number in each queue is available, while the Round-Robin (RR), followed by FIFO, is shown to be the optimal policy when it is only known that the queues were initially in the same state. [24] proves that RR/FIFO is optimal with the absence of queue length information if the job sizes have a non-decreasing hazard function. The RR results were later generalized in [23]. Whitt [26], on the other hand, provides several counterexamples where JSQ/FIFO policy fails. Crovella et al.[6] and Harchol-Balter et al. [14] assume that the dispatcher is aware of the size of a new job, but not of the state of the FIFO queues, and propose policies based on job size intervals (e.g., short jobs to one queue, and the rest to another). Feng et al. [10] later showed that such a policy is the optimal size-aware state-independent dispatching policy for homogeneous servers.

Also the MDP framework lends itself to dispatching problems. Krishnan [22] has utilized it in the context of parallel M/M/s-FIFO servers so as to minimize the mean sojourn time, similarly as Aalto and Virtamo [3] for the traditional M/M/1 queue. Recently, FIFO, LIFO, SPT and SRPT queues were analyzed in [19] with a general service time distribution. Similarly, PS is considered in [21], [20]. The key idea with the above work is to start by an arbitrary state-independent policy, and then carry out FPI step utilizing the value functions (relative values of states).

II. M/G/1 Queue and Slowdown

In this section we consider a single M/G/1 queue with arrival rate λ. The service requirements are i.i.d. random variables $X_i \sim X$ with a general distribution.

We define the *slowdown* of a job as a ratio of the sojourn time T to the service requirement X,

$$\gamma \triangleq \frac{T}{X},$$

where one is generally interested in its mean value $\mathrm{E}[\gamma]$. Also other similar definitions exist. In the context of distributed computing, Harchol-Balter defines the slowdown as the "wall-time" divided by the CPU-time in [15]. Another common convention is to consider the ratio of the waiting time to the processing time (see, e.g., [13]), which is a well-defined quantity for non-preemptive systems. These different definitions, however, are essentially the same.

For non-preemptive work conserving policies, the sojourn time in queue comprises the initial waiting time W and the consequent processing time X. Thus, the mean slowdown is

$$\mathrm{E}[\gamma] = \mathrm{E}[\frac{W+X}{X}] = 1 + \mathrm{E}[W/X].$$

Waiting time with FIFO is independent of the job size X, and with aid of Pollaczek-Khinchin formula one obtains [13],

$$\mathrm{E}[\gamma] = 1 + \mathrm{E}[W] \cdot \mathrm{E}[X^{-1}] = 1 + \frac{\lambda\, \mathrm{E}[X^2]}{2(1-\rho)} \cdot \mathrm{E}[X^{-1}], \quad (1)$$

which underlines the fact that the mean slowdown may be infinite for a stable queue as $\mathrm{E}[X^{-1}] = \infty$ for many common job size distributions, e.g., for an exponential distribution.

The mean slowdown in an M/G/1 queue with preemptive LIFO and PS disciplines is a constant for all job sizes,

$$\mathrm{E}[\gamma \mid X = x] = \frac{1}{1-\rho}. \quad (2)$$

Comparison of (1) and (2) immediately gives:

Corollary 1: FIFO is a better scheduling discipline than LIFO in an M/G/1 queue with respect to slowdown iff

$$2\,\mathrm{E}[X] > \mathrm{E}[X^2] \cdot \mathrm{E}[X^{-1}]. \quad (3)$$

A. Shortest-Processing-Time-Product (SPTP)

In [30], Yang and de Veciana introduced the *shortest-processing-time-product* (SPTP) scheduling discipline, which will serve the job i^* such that

$$i^* = \arg\min_i \Delta_i^* \Delta_i,$$

where Δ_i^* and Δ_i denote, respectively, the initial and remaining service requirement of job i. Yang and de Veciana were able to show that SPTP is optimal with respect to the mean slowdown $\mathrm{E}[\gamma]$ in a transient system where all jobs are available at time 0 and no new jobs arrive thereafter (i.e., a *myopic* approach). SPTP can be seen as the counterpart of the SRPT[3] when instead of minimizing the mean sojourn time, one is interested in minimizing the mean slowdown.

We argue below that SPTP is the optimal scheduling discipline also in the corresponding dynamic system with Poisson arrivals. That is, we show that SPTP is optimal with respect to the slowdown in the M/G/1 queue. Our proof is based on the Gittins index approach.

[3]Note that defining an index policy with indices Δ_i, Δ_i^* and $\sqrt{\Delta_i \Delta_i^*}$ gives SRPT, SPT and SPTP, respectively, where $\sqrt{\Delta_i \Delta_i^*}$ corresponds to the geometric mean of Δ_i and Δ_i^*.

B. *Gittins Index*

Consider an M/G/1 queue with *multiple* job classes k, $k = 1, \ldots, K$. Let λ_k denote the class-k arrival rate, and $p_k = \lambda_k/\lambda$ the probability that an arriving job belongs to class k. In addition, let X_k denote the generic service time related to class k. The total load is denoted by $\rho = \lambda_1 \mathrm{E}[X_1] + \ldots + \lambda_K \mathrm{E}[X_K]$. We assume that $\rho < 1$. The *Gittins index* [11], [2] for a class-k job with attained service a is defined by

$$G_k(a) = \sup_{\delta > 0} w_k \, \frac{\mathrm{P}\{X_k - a \le \delta \mid X_k > a\}}{\mathrm{E}[\min\{X_k - a, \delta\} \mid X_k > a]},$$

where w_k is the holding cost rate related to class k. Note that, in addition to the attained service a, the Gittins index is based on the (class-specific) distribution of the service time, but not on the service time itself. The *Gittins index policy* will serve the job i^* such that

$$i^* = \arg\max_i G_{k_i}(a_i),$$

where k_i refers to the class and a_i to the attained service of job i. Let T_k denote the sojourn time of a class-k job. We recall the following optimality result proved by Gittins [11, Theorem 3.28].

Theorem 1: Consider an $M/G/1$ multi-class queue. The Gittins index policy minimizes the mean holding costs,

$$\sum_k p_k w_k \mathrm{E}[T_k],$$

among the non-anticipating scheduling policies.

The non-anticipating policies are only aware of the attained service times a_i of each job i and the service time distributions (but not on the actual service times). So, in general, non-anticipating policies do not know the remaining service times implying that, e.g., SPTP does not belong to non-anticipating disciplines. However, there is one exception: If the service times X_k are deterministic, $\mathrm{P}\{X_k = x_k\} = 1$, then the remaining service times $x_k - a_k$ are available for the non-anticipating policies (with probability 1). In such a case, all policies are non-anticipating.

C. *Optimality of SPTP*

Consider now an M/G/1 queue with a *single* class and load $\rho < 1$. Let X and T denote, respectively, the generic service and sojourn times of a job. In addition, let $\tau(x)$ denote the conditional mean sojourn time of a job with service time x,

$$\tau(x) \triangleq \mathrm{E}[T \mid X = x].$$

Assume first that the support of the service time distribution is finite. In other words, there are $x_1 < \ldots < x_K$ such that $\sum_k \mathrm{P}\{X = x_k\} = 1$. The following result is an immediate consequence of Theorem 1 as soon as we associate class k with the jobs with the same service time requirement x_k. Note that the Gittins index is now clearly given by

$$G_k(a) = \frac{w_k}{x_k - a}$$

with the optimal δ equal to $x_k - a$.

Corollary 2: Consider an $M/G/1$ queue for which the support of the service time distribution is finite. Among all scheduling policies, the mean holding costs,

$$\sum_k \mathrm{P}\{X = x_k\} w_k \tau(x_k),$$

are minimized by the policy that will serve job i^* such that

$$i^* = \arg\min_i \frac{\Delta_i}{w_{k(i)}}$$

where $k(i)$ and Δ_i denote the class and the remaining service requirement of job i.

In particular, if we choose $w_k = 1/x_k$, we see that the mean slowdown is minimized by SPTP:

Corollary 3: Consider an $M/G/1$ queue for which the support of the service time distribution is finite. SPTP minimizes the mean slowdown $\mathrm{E}[\gamma]$ among all scheduling policies.

Since any service time distribution can be approximated with an arbitrary precision by discrete service time distributions, we expect that the above result holds also for general service time distributions. Note also that the choice $w_k = 1$, for all k, results in the well-known optimality result of SRPT with respect to the mean sojourn time.

III. Value Functions for M/G/1

In this section we analyze a single M/G/1 queue in isolation with FIFO, LIFO, and SPTP scheduling disciplines and derive expressions for the size-aware value functions *with respect to arbitrary job specific holding costs*. The corresponding results for SPT and SRPT are also given in the Appendix. As special cases, one trivially obtains the value functions with respect to the mean sojourn time and the mean slowdown. Thus, these results generalize the corresponding results in [19], where the size-aware value functions with respect to sojourn time are given (except for SPTP).

Holding cost model: Consider an M/G/1 queue with an arbitrary but fixed work conserving scheduling discipline and load $\rho < 1$. Each existing task incurs costs at some *task specific holding cost rate* denoted by B_i for job i. Assume that B_i are i.i.d. random variables, $B_i \sim B$. However, B_i may depend on the corresponding service requirement X_i. Both the service requirement and the holding cost become known upon arrival.

The cumulative cost during $(0, t)$ for an initial state $\mathbf{z}$ is

$$V_{\mathbf{z}}(t) \triangleq \int_0^t \sum_{i \in \mathcal{N}_{\mathbf{z}}(s)} B_i \, ds,$$

where $\mathcal{N}_{\mathbf{z}}(s)$ denotes the set of tasks present in the system at time s. Similarly, the long-term mean cost rate is

$$r_{\mathbf{z}} \triangleq \lim_{t \to \infty} \frac{1}{t} \mathrm{E}[V_{\mathbf{z}}(t)] = \lim_{t \to \infty} \frac{1}{t} \mathrm{E}[\int_0^t \sum_{i \in \mathcal{N}_{\mathbf{z}}(s)} B_i \, ds],$$

which for an ergodic Markovian system reads

$$r = \lim_{t \to \infty} \mathrm{E}[\sum_{i \in \mathcal{N}_{\mathbf{z}}(t)} B_i].$$

The value function $v_{\mathbf{z}}$ is defined as the expected deviation from the mean cost rate in infinite time-horizon,

$$v_{\mathbf{z}} \triangleq \lim_{t\to\infty} \mathrm{E}[V_{\mathbf{z}}(t) - r\cdot t].$$

A value function characterizes how expensive it is to start from each state. In our setting, it enables one to compute the expected cost of admitting a job with size x and holding cost rate b to a queue, which afterwards behaves as an M/G/1 queue with the given scheduling discipline:

$$\omega_{\mathbf{z}}(x,b) \triangleq v_{\mathbf{z}\oplus(x,b)} - v_{\mathbf{z}}, \tag{4}$$

where $\mathbf{z}\oplus(x,b)$ denotes the state resulting from adding a new job (x,b) in state $\mathbf{z}$. In Section IV we will carry out the FPI step in the context of dispatching problem, for which a corresponding value function is a prerequisite.

Size dependent holding cost: In general, the holding cost rate b can be arbitrary, e.g., a class-specific i.i.d. random variable. However, in two important special cases, mean sojourn time and mean slowdown, it depends solely on the service requirement x, $b = c(x)$:

mean sojourn time:	$c(x) = 1$
mean slowdown:	$c(x) = 1/x$

In equilibrium, the average cost *per job* is $\mathrm{E}[c(X)\cdot T]$. Using Little's result, we have for the *mean cost rate*,

$$r = \lambda\,\mathrm{E}[c(X)\cdot T] = \begin{cases} \lambda\,\mathrm{E}[T] = \mathrm{E}[N], & \text{for } c(x)=1,\\ \lambda\,\mathrm{E}[\gamma], & \text{for } c(x)=1/x. \end{cases}$$

where $\mathrm{E}[N]$ denotes the mean number in the system.

A. M/G/1-FIFO Queue

Next we derive the size-aware value function for an M/G/1-FIFO queue with respect to arbitrary job specific holding costs. To this end, let $\mathbf{z} = ((\Delta_1,b_1);\ldots;(\Delta_n,b_n))$ denote the remaining service requirements (measured in time) Δ_i and the corresponding *holding cost rates* b_i at state $\mathbf{z}$. The total rate at which costs are accrued at given state $\mathbf{z}$ is thus the sum $\sum_i b_i$. Job 1 is currently receiving service and job n is the latest arrival. The total backlog is denoted by $u_{\mathbf{z}}$,

$$u_{\mathbf{z}} = \sum_{i=1}^{n} \Delta_i.$$

Proposition 1: For the size-aware relative value in an M/G/1-FIFO queue with respect to arbitrary job specific holding costs it holds that

$$v_{\mathbf{z}} - v_0 = \sum_{i=1}^{n}\left(b_i \sum_{j=1}^{i}\Delta_j\right) + \frac{\lambda\cdot\mathrm{E}[B]}{2\,(1-\rho)}u_{\mathbf{z}}^2 \tag{5}$$

where $\mathrm{E}[B]$ is the mean holding cost rate for all later jobs.

Proof: We compare two systems with the same arrival patterns, System 1 initially in state $\mathbf{z}$ and System 2 initially empty. The two systems behave equivalently after System 1 becomes empty. The cost incurred by the current n jobs, present only in System 1, is already fixed as

$$h_1 = \sum_{i=1}^{n}\left(b_i\sum_{j=1}^{i}\Delta_j\right).$$

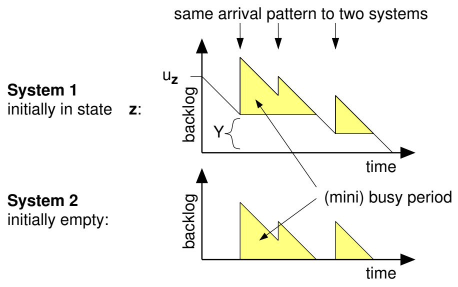


Fig. 2. Derivation of the value function in an M/G/1-FIFO queue.

The later arriving jobs encounter a longer waiting time in System 1. The key observation here is that these jobs experience an additional delay of Y in System 1 when compared to System 2, as is illustrated in Fig. 2. Otherwise the sojourn times are equal. On average $\lambda u_{\mathbf{z}}$ (mini) busy periods occur before System 1 becomes empty. These busy periods are independent and on average $1/(1-\rho)$ jobs are served during each of them. The average additional waiting time is $\mathrm{E}[Y] = u_{\mathbf{z}}/2$. Therefore, the later arriving jobs incur on average

$$h_2 = \frac{\lambda\cdot\mathrm{E}[B]}{2\,(1-\rho)}u_{\mathbf{z}}^2$$

higher holding cost in System 1 than in System 2. Total cost difference is $h_1 + h_2$, which completes the proof. ■

Corollary 4: The mean cost in terms of sojourn time due to accepting a job with size x to a size-aware M/G/1-FIFO queue initially at state $\mathbf{z}$ is

$$\omega_{\mathbf{z}}(x) = x + u_{\mathbf{z}} + \frac{\lambda}{2(1-\rho)}(2\,u_{\mathbf{z}}\,x + x^2).$$

Corollary 5: The mean cost in terms of slowdown due to accepting a job with size x to a size-aware M/G/1-FIFO queue initially at state $\mathbf{z}$ is

$$\omega_{\mathbf{z}}(x) = 1 + \frac{u_{\mathbf{z}}}{x} + \frac{\lambda\cdot\mathrm{E}[X^{-1}]}{2(1-\rho)}(2u_{\mathbf{z}}x + x^2). \tag{6}$$

In both cases, it is implicitly assumed that the question about the admittance is a one-time operation and the future behavior of the queue is according to the standard M/G/1-FIFO. The proofs follow trivially from (4) and (5).

B. M/G/1-LIFO Queue

Next we derive the size-aware value function for the LIFO scheduling discipline. To this end, let us denote the state of the system with n jobs by a vector $\mathbf{z} = ((\Delta_1,b_1);\ldots;(\Delta_n,b_n))$, where Δ_i denotes the remaining service requirement (measured in time) and b_i the *holding cost rate* of job i. Job 1 is the latest arrival and currently receiving service (if any), i.e., without new arrivals the jobs are processed in the natural order: $1, 2, \ldots, n$.

Proposition 2: For the size-aware relative value in a preemptive M/G/1-LIFO queue with respect to arbitrary job specific holding cost it holds that

$$v_{\mathbf{z}} - v_0 = \frac{1}{1-\rho} \sum_{i=1}^{n} \left(b_i \sum_{j=1}^{i} \Delta_j \right). \tag{7}$$

Proof: We compare System 1 initially in state $\mathbf{z}$ and System 2 initially empty. The current state bears no meaning for the future arrivals with preemptive LIFO, and thus the difference in the expected costs is equal to the cost the current n jobs in System 1 incur:

$$v_{\mathbf{z}} - v_0 = \sum_{i=1}^{n} b_i \, \mathrm{E}[R_i],$$

where R_i denotes the remaining sojourn time of job i [19],

$$\mathrm{E}[R_i] = \frac{\sum_{j=1}^{i} \Delta_j}{1-\rho},$$

which completes the proof. ■

For the mean sojourn time, the holding cost rate is constant $b_i = 1$ and a sufficient state description is $(\Delta_1, \ldots, \Delta_n)$.

Corollary 6: The cost in terms of sojourn time due to accepting a job with size x to a size-aware preemptive M/G/1-LIFO queue at state $\mathbf{z}$ is

$$\omega_{\mathbf{z}}(x) = \frac{1}{1-\rho}(n+1)x.$$

For the slowdown, the initial service requirements define the holding costs and a sufficient state description is $\mathbf{z} = ((\Delta_1, \Delta_1^*), .., (\Delta_1, \Delta_1^*))$, where Δ_i and Δ_i^* denote the remaining and initial service requirement of job i, so that the holding cost of job i is $b_i = 1/\Delta_i^*$:

Corollary 7: The cost in terms of slowdown due to accepting a job with size x to a size-aware preemptive M/G/1-LIFO queue at state $\mathbf{z}$ is

$$\omega_{\mathbf{z}}(x) = \frac{1}{1-\rho} \left(1 + \sum_{i=1}^{n} x/\Delta_i^* \right). \tag{8}$$

The proofs follow trivially from (7) and from definition (4). Note that $\omega_{\mathbf{z}}(x)$ with (preemptive) LIFO does not depend on the remaining service times Δ_i due to the preemption.

C. M/G/1-SPTP Queue

Next we derive the corresponding size-aware value function for the SPTP scheduling discipline. The state description for SPTP is $\mathbf{z} = ((\Delta_1, \Delta_1^*, b_1), \ldots, (\Delta_n, \Delta_n^*, b_n))$, where, without loss of generality, we assume a decreasing priority order, $\Delta_i \Delta_i^* < \Delta_{i+1}\Delta_{i+1}^*$, i.e., the job 1 (if any) is currently receiving service. Then, $u_{\mathbf{z}}(h)$ denotes the amount of work with a higher priority than h,

$$u_{\mathbf{z}}(h) \triangleq \sum_{i:\Delta_i \Delta_i^* < h} \Delta_i.$$

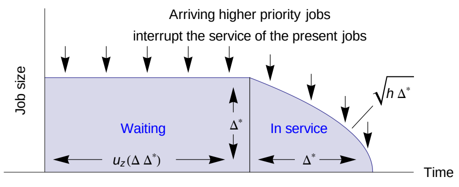


Fig. 3. Remaining sojourn time of an (Δ, Δ^*)-job in an M/G/1-SPTP queue.

Let $\lambda(x)$, $m(x)$ and $\rho(x)$ denote the arrival rate, mean job size and offered load due to jobs shorter than x,

$$\begin{array}{ll} \lambda(x) & \triangleq \lambda \, \mathrm{P}\{X < x\}, \\ m(x) & \triangleq \mathrm{E}[X \mid X < x], \\ \rho(x) & \triangleq \lambda \int_0^x t\, f(t)\, dt, \end{array} \tag{9}$$

where $f(x)$ denotes the job size pdf.

Lemma 1: The mean remaining sojourn time of a (Δ, Δ^*)-job in an M/G/1-SPTP queue initially in state $\mathbf{z}$ is given by

$$\mathrm{E}[R_{\mathbf{z}}(\Delta, \Delta^*)] = \frac{u_{\mathbf{z}}(\Delta\Delta^*)}{1-\rho(\sqrt{\Delta\Delta^*})} + \frac{2}{\Delta^*} \int_0^{\sqrt{\Delta\Delta^*}} \frac{x\,dx}{1-\rho(x)}. \tag{10}$$

Proof: Let k denote the (Δ, Δ^*)-job, whose remaining sojourn time depends on the initial and later arriving higher priority work. We can assume that the latter are served immediately according to LIFO, thus triggering mini busy periods. Let $h = h(t)$ denote the remaining service time of job k at (virtual) time t, where we have omitted these mini busy periods from the time axis. During $0 < t < u_{\mathbf{z}}(\Delta\Delta^*)$, job k is waiting and $h = \Delta$, but as the service begins $h \to 0$ linearly. The later arriving higher priority jobs shorter than $\sqrt{h\Delta^*}$ constitute an inhomogeneous Poisson process with rate $\lambda(\sqrt{h\Delta^*})$, as illustrated in Fig. 3. The mean duration of a mini busy period is (cf. the mean busy period in M/G/1),

$$D(h) \triangleq \frac{m(\sqrt{h\,\Delta^*})}{1-\rho(\sqrt{h\,\Delta^*})}.$$

The mean waiting time before job k receives service for the first time is $u_{\mathbf{z}}(\Delta\Delta^*) + \lambda(\sqrt{h\Delta^*})\, u_{\mathbf{z}}(\Delta\Delta^*) \cdot D(\Delta)$, which gives the first term in (10).

The service time Δ and the additional delays during the service are on average

$$\Delta + \int_0^{\Delta} \lambda(\sqrt{h\,\Delta^*}) \cdot D(h)\, dh;$$

refer to the "in service" region of Fig. 3. Change of integration variable, $x = \sqrt{h\,\Delta^*}$ then gives the second term in (10), which completes the proof. ■

Even though SPTP explicitly tries to minimize the mean slowdown, we derive next a general expression for the value function with arbitrary job specific holding costs:

Proposition 3: For the size-aware relative value in an M/G/1-SPTP queue with respect to arbitrary job specific

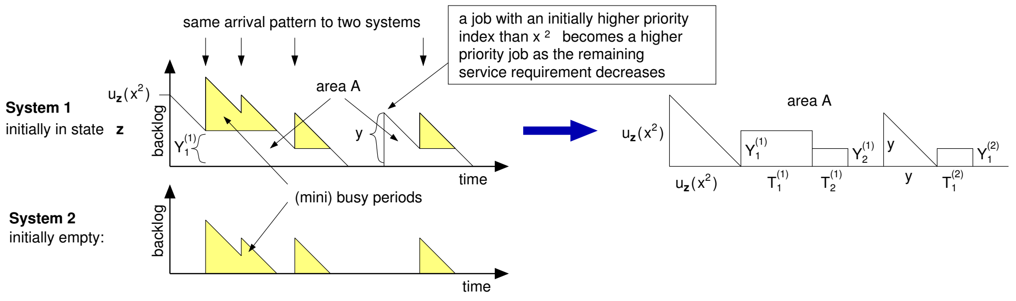


Fig. 4. Derivation of the value function for an M/G/1-SPTP queue with respect to slowdown.

holding costs it holds that

$$v_{\mathbf{z}}-v_0 = \sum_{i=1}^{n} b_i\left(\frac{\sum_{j=1}^{i-1}\Delta_j}{1-\rho(\tilde{\Delta}_i)} + \frac{2}{\Delta_i^*}\int_0^{\tilde{\Delta}_i}\frac{x\,dx}{1-\rho(x)}\right) + \frac{\lambda}{2}\sum_{i=0}^{n}\Bigg[(\sum_{j=1}^{i}\Delta_j)^2\int_{\tilde{\Delta}_i}^{\tilde{\Delta}_{i+1}}\frac{b(x)\,f(x)}{(1-\rho(x))^2}\,dx + \sum_{j=i+1}^{n}(\Delta_j^*)^{-2}\int_{\tilde{\Delta}_i}^{\tilde{\Delta}_{i+1}}\frac{x^4\,b(x)\,f(x)}{(1-\rho(x))^2}\,dx\Bigg] \tag{11}$$

where $b(x) \triangleq \mathrm{E}[B \mid X = x]$ and

$$\tilde{\Delta}_i = \begin{cases} 0, & i=0 \\ \sqrt{\Delta_i\Delta_i^*}, & i=1,\ldots,n \\ \infty, & i=n+1. \end{cases}$$

Proof: The relative value comprises the mean holding cost h_1 accrued by the current n tasks in state $\mathbf{z}$,

$$\mathbf{z} = ((\Delta_1, \Delta_1^*, b_1), \ldots, (\Delta_n, \Delta_n^*, b_n)),$$

and the difference in costs accrued by the later arriving jobs, h_2, $v_{\mathbf{z}} - v_0 = h_1 + h_2$. The first summation over n gives h_1, i.e., it follows from multiplying (10) with the job specific holding cost b_i and adding over all i.

The latter integral corresponds to h_2 where the accrued costs are conditioned on the size x of an arriving job. The corresponding arrival rate is $\lambda f(x)\,dx$. The mean difference in sojourn time experienced by an arriving job with size x is accrued during the initial waiting time – once a job enters the service for the first time its mean remaining sojourn time is the same in both systems. The initial waiting time is a function of higher priority workload upon arrival, denoted by $U_{\mathbf{z}}(x^2,t)$ for an initial state $\mathbf{z}$ at arrival time t. In particular, the mean (initial) waiting time is simply $\mathrm{E}[U_{\mathbf{z}}(x^2,t)]/(1-\rho(x))$, which gives

$$h_2 = \lambda\int_0^\infty \frac{f(x)}{1-\rho(x)}\,(\mathrm{E}[U_{\mathbf{z}}(x^2,t) - U_0(x^2,t)])\,dx.$$

Next we refer to Fig. 4 and observe that $U_{\mathbf{z}}(x^2,t) - U_0(x^2,t)$ in the integrand corresponds to area A. This area consists of one $u_{\mathbf{z}}(x^2)$ triangle, followed by N_0 rectangles, and similar sequences starting with a x^2/Δ_i^* triangle for $\{i : \Delta_i\Delta_i^* > x^2\}$, each followed by N_i rectangles. The N_i are random variables corresponding to the number of mini busy periods during the service time of a particular triangle. The first triangle corresponds to the initially higher priority workload $u_{\mathbf{z}}(x^2)$, and the latter to the jobs with initially lower priority, i.e., to jobs i with $\Delta_i\Delta_i^* > x^2$. At some point in time, the remaining service requirement of such a job i decreases to $y_i = x^2/\Delta_i^*$ and its priority index drops below x^2. For a sequence starting with a y-triangle, the number of mini busy periods (rectangles) obeys Poisson distribution with mean $\lambda(x)\,y$. The height of a rectangle is uniformly distributed in $(0, y)$ having the mean $y/2$ (property of Poisson process), while the width corresponds to the duration of a busy period in a work conserving M/G/1 queue having the mean $m(x)/(1-\rho(x))$. Thus, the mean total area is

$$\frac{y^2}{2} + \lambda(x)y\cdot\frac{y}{2}\cdot\frac{m(x)}{1-\rho(x)} = \frac{y^2}{2(1-\rho(x))}.$$

Therefore,

$$h_2 = \frac{\lambda}{2}\int_0^\infty \frac{b(x)\,f(x)}{(1-\rho(x))^2}\left(u_{\mathbf{z}}(x^2)^2 + \sum_{i:\Delta_i\Delta_i^*>x^2}\left(\frac{x^2}{\Delta_i^*}\right)^2\right)dx,$$

where $b(x) = \mathrm{E}[B \mid X = x]$ factor in the numerator corresponds to the mean holding cost of an x-job. For each interval $x \in (\tilde{\Delta}_i, \tilde{\Delta}_{i+1})$, $u_{\mathbf{z}}(x^2) = \sum_{j=1}^{i-1}\Delta_j$ and

$$\sum_{j:\Delta_j\Delta_j^*>x^2}\left(\frac{x^2}{\Delta_j^*}\right)^2 = x^4\sum_{j=i+1}^{n}(\Delta_j^*)^{-2},$$

and integration in $n+1$ parts completes the proof. ■

Recall that with respect to the mean sojourn time, $b_i = 1$ and $b(x) = 1$, while for the slowdown criterion, $b_i = 1/\Delta_i^*$ and $b(x) = 1/x$. Hence, (11) allows one to compute the mean cost $\omega_{\mathbf{z}}(x)$ due to accepting a given job in terms of sojourn time or slowdown. We omit the explicit expression for $\omega_{\mathbf{z}}(x)$ for brevity.

The corresponding value functions for SPT and SRPT are derived in the Appendix. The numerical evaluation of the value function for SPTP, SPT and SRPT is not as unattractive as it first may seem. Basically one needs to be able to compute integrals of form

$$F_k(x) = \int_0^x \frac{t^k\,b(t)\,f(t)}{(1-\rho(t))^2}\,dt, \text{ and } G_d(x) = \int_0^x \frac{t^d}{1-\rho(t)}\,dt,$$

where $k = 0, 2, 4$ and $d = 0, 1$ depending on the discipline. Thus, e.g., a suitable interpolation of the $F_k(x)$ and $G_d(x)$ enables on-line computation of the value function.

IV. Dispatching Problem

Next we utilize the results of Section III in the dispatching problem with parallel servers and focus solely on minimizing the mean slowdown. The dispatching system illustrated in Fig. 1 comprises m servers with service rates $\nu_1, \ldots, \nu_m$. Jobs arrive according to a Poisson process with rate λ and their service requirements are i.i.d. random variables with a general distribution. The jobs are served according to a given scheduling discipline (e.g., FIFO) in each server.

The slowdown for an isolated queue was defined as the sojourn time T divided by the service requirement X [17] (both measured in time), $\gamma = T/X$. However, as we consider heterogeneous servers with rates ν_i, the service requirement X of size Y job (measured, e.g., in bytes) is no longer unambiguous but a server specific quantity,

$$X_i = Y/\nu_i.$$

Therefore, for a dispatching system we compare the sojourn time T to the hypothetical service time if all capacity could be assigned to process a given job [7],

$$\gamma^* \triangleq \frac{T}{Y/\sum_i \nu_i}.$$

The relationship between the queue i specific slowdown γ and the system wide slowdown γ^* is $\gamma^* = \gamma \cdot (\sum_j \nu_j)/\nu_i$.

A. Random Dispatching Policies

The so-called Bernoulli splitting assigning jobs independently in random using probability distribution $(p_1, \ldots, p_m)$ offers a good state-independent basic dispatching policy. Due to Poisson arrivals, each queue also receives jobs according to a Poisson process with rate $p_i\,\lambda$.

Definition 1 (RND-ρ): The *RND-ρ dispatching policy* balances the load equally by setting $p_i = \nu_i / \sum_j \nu_j$.

As an example, the mean slowdown in a preemptive LIFO or PS queue with RND-ρ policy is,

$$\mathrm{E}[\gamma^*] = \Big(\sum_j \nu_j\Big) \frac{m}{\sum_j \nu_j - \lambda\,\mathrm{E}[Y]},$$

where the denominator corresponds to the excess capacity. Thus, with RND-ρ and LIFO queues, the slowdown criterion is m times higher when m servers are used instead of a single fast one irrespectively of the service rate distribution ν_i.

For identical servers, $\nu_1 = \nu_2 = \ldots = \nu_m$, the RND-$\rho$ dispatching policy reduces to RND-U:

Definition 2 (RND-U): The RND-U dispatching policy assigns jobs randomly in uniform using $p_i = 1/m$.

RND-U is obviously the optimal random policy in case of identical servers.

In general, the optimal splitting probabilities depend on the service time distribution and the scheduling discipline. For the preemptive LIFO (and PS) server systems the mean slowdown with a random dispatching policy is given by

$$\mathrm{E}[\gamma^*] = \Big(\sum_j \nu_j\Big) \sum_{i=1}^{m} \frac{p_i}{\nu_i - p_i \cdot \lambda\,\mathrm{E}[Y]},$$

where $\sum_j \nu_j$ is a constant that can be neglected when optimizing the p_i.

Definition 3 (RND-opt): The optimal random dispatching policy for LIFO/PS queues, referred to as RND-opt, splits the incoming tasks using the probability distribution,

$$p_i = \frac{\nu_i - \sqrt{\nu_i}\,G}{\lambda\,\mathrm{E}[Y]}, \quad \text{where } G = \frac{\sum_i \nu_i - \lambda\,\mathrm{E}[Y]}{\sum_i \sqrt{\nu_i}}.$$

The result is easy to show with the aid of Lagrange multipliers. We note that when some servers are too slow, the above gives infeasible values for some p_i, in case of which one simply excludes the slowest server from the solution and re-computes a new probability distribution. This is repeated until a feasible solution is found. A more explicit formulation is given in [5], [18]. RND-opt is insensitive to the job size distribution and optimal only for LIFO and PS.

Pollaczek-Khinchin mean value formula enables one to analyze FIFO queues and, e.g., to write an expression for the mean slowdown with RND-ρ. Also the optimal splitting probabilities can be computed numerically for an arbitrary job size distribution. According to (1), the mean slowdown in each queue depends on the mean waiting time $\mathrm{E}[W_i]$ and $\mathrm{E}[X_i^{-1}]$, where the latter is assumed to exist and to be finite. Hence, the optimal probability distribution with respect to slowdown is the same as with the mean sojourn time.

Similarly, the mean sojourn time in SPT and SRPT queues is known which allows a numerical optimization of the splitting probabilities. For simplicity, we consider only RND-ρ and RND-opt in this paper as compact closed form expressions for the p_i are only available for these.

B. SITA-E Dispatching Policy

With FIFO queues, a state-independent dispatching policy known as the size-interval-task-assignment (SITA) has proven to be efficient especially with heavy-tailed job size distributions [6], [14], [4]. The motivation behind SITA is to segregate the long jobs from the short ones. Reality, however, is more complicated and segregating the jobs categorically can also give suboptimal results [16]. Here we assume $\nu_1 \geq \nu_2 \geq \ldots \geq \nu_m$ and a continuous job size distribution with pdf $f(x)$.

Definition 4 (SITA): A SITA policy is defined by disjoint job size intervals $\{(\xi_0, \xi_1],\ (\xi_1, \xi_2],\ \ldots,\ (\xi_{m-1}, \xi_m]\}$, and assigns a job with size x to server i iff $x \in (\xi_{i-1}, \xi_i]$.

Without loss of generality, one can assume that $\xi_0 = 0$ and $\xi_m = \infty$. Note that in contrast to random policies, SITA assumes that the dispatcher is aware of the size of the new job. In this paper, we limit ourselves to SITA-E, where E stands for *equal load.* That is, the size intervals are chosen in such a way that the load is balanced between the servers.

Definition 5 (SITA-E): With SITA-E dispatching policy the thresholds ξ_i are defined in such a way that

$$\frac{1}{\nu_i}\int_{\xi_{i-1}}^{\xi_i} x\,f(x)\,dx = \frac{1}{\nu_j}\int_{\xi_{j-1}}^{\xi_j} x\,f(x)\,dx, \qquad \forall\, i,j.$$

Thus, similarly as RND-ρ, also the SITA-E policy is insensitive to the arrival rate λ.

C. Improved Dispatching Policies

The important property the above state-independent dispatching policies have is that the arrival process to each queue is a Poisson process. Consequently, the value functions derived earlier allow us to quantify the value function of the whole system. With a slight abuse of notation,

$$v_{\tilde{\mathbf{z}}} - v_{\mathbf{0}} = \sum_{i=1}^{m} (v_{\mathbf{z}_i}^{(i)} - v_0^{(i)}),$$

where $v_{\mathbf{z}_i}^{(i)}$ denotes the relative value of queue i in state $\mathbf{z}_i$, and $\tilde{\mathbf{z}} = (\mathbf{z}_1, \ldots, \mathbf{z}_m)$ the state of the whole system.

Policy Improvement by Role Switching: One interesting opportunity to utilize the relative values is to *switch* the roles of two queues [19]. Namely, with an arbitrary state-independent policy one can switch the input processes of any two identical servers at any moment, and effectively end up to a new state of the same system. Despite its limitation (identical servers), *switching* is an interesting policy improvement method that requires little additional computation. The switch should only be carried out when the new state has lower expected future costs, i.e. when for some $i \neq j$, the rates are equal, $\nu_i = \nu_j$, and

$$v_{(\mathbf{z}_1,..,\mathbf{z}_i,..\mathbf{z}_j,..,\mathbf{z}_m)} < v_{(\mathbf{z}_1,..,\mathbf{z}_j,..\mathbf{z}_i,..,\mathbf{z}_m)}.$$

Carrying out this operation whenever a new job has arrived to the system reduces the state-space by *removing* such states for which a better alternative to continue exists. Formally, let $\pi(\tilde{\mathbf{z}})$ denote the set of feasible permutations of the queues' roles, where the input processes between identical servers are switched. Then, the optimal state-space reduction, implicitly defining a new policy, is given by

$$\tilde{\mathbf{z}} \leftarrow \underset{\tilde{\mathbf{z}}' \in \pi(\tilde{\mathbf{z}})}{\operatorname{argmin}}\, v_{\tilde{\mathbf{z}}'}.$$

To elaborate this, consider a RND-ρ/LIFO system. In this case, (7) implies that switching the roles of any two identical queues makes no difference to the relative value.

In contrast, with FIFO discipline the interesting quantity from (5) for two identical servers i and j is identified to be

$$C_{ij} \triangleq \frac{\lambda_i\,\mathrm{E}[B_i]}{1-\rho_i} \cdot u_{\mathbf{z}_j}^2.$$

Switching the input processes between queues i and j leads to a state with a lower relative value if $C_{ii} + C_{jj} > C_{ij} + C_{ji}$. Again, with RND-$\rho$ the factor $\lambda_i\,\mathrm{E}[B_i]/(1-\rho_i)$ is a constant for any two identical servers and switching provides no gain. However, with SITA-E, even though the denominator $1-\rho_i$ is a constant, the numerator is not. Let X_i denote a job size in queue i. Then $\mathrm{E}[X_i] < \mathrm{E}[X_{i+1}]$, and therefore $\lambda_i > \lambda_{i+1}$. Moreover, $\mathrm{E}[1/X_i] > \mathrm{E}[1/X_{i+1}]$, yielding

$$\lambda_i\,\mathrm{E}[1/X_i] > \lambda_{i+1}\,\mathrm{E}[1/X_{i+1}].$$

Consequently, with the *SITA-E with switch* policy, the optimal permutation, after inserting a new job to a queue, is the one with increasing backlogs for identical servers:

Definition 6 (SITA-Es): SITA-E with switch dispatching policy behaves similarly as SITA-E with a distinction that after each task assignment the queues are permutated in such a way that $u_{\mathbf{z}_i} \leq u_{\mathbf{z}_{i+1}}$ for all identical servers i and $i+1$.

Policy Improvement by FPI: The *first-policy-iteration* (FPI) is a general method of the MDP framework to improve any given policy. We apply it to the dispatching problem [19]. Suppose that the scheduling discipline in each queue is fixed and that a state-independent basic dispatching policy would assign a new job to some queue. Given the relative values and the expected cost associated with accepting the job to each queue, we can carry out FPI: we deviate from the default action if the expected cost is smaller with some other action, thereby decreasing the expected cumulative costs in infinite time horizon.

Let λ_i denote the arrival rate to queue i according to the basic dispatching policy, and Y_i the corresponding job size. With a state-independent policy, accepting a job to queue i does not affect the future behavior of the other queues. Thus, the cost of assigning a job with size y to queue i is

$$\omega_{\tilde{\mathbf{z}}}(y,i) = \omega_{\mathbf{z}_i}^*(y).$$

where $\omega_{\mathbf{z}_i}^*(y)$ is the mean admittance cost of a job with size y (measured, e.g., in bytes) to queue i, where its service requirement (measured in time) would be y/ν_i. For slowdown, we have an elementary relation

$$\omega_{\mathbf{z}_i}^*(y) = \frac{\sum_j \nu_j}{\nu_i} \cdot \omega_{\mathbf{z}_i}(y/\nu_i).$$

For *FIFO queues*, (6) gives

$$\omega_{\mathbf{z}_i}^*(y) = \big(\sum_j \nu_j\big)\left(\frac{1}{\nu_i} + \frac{u_{\mathbf{z}_i}}{y} + \frac{\lambda_i\,\mathrm{E}[Y_i^{-1}]}{2} \cdot \frac{2u_{\mathbf{z}_i}y + y^2/\nu_i}{\nu_i - \lambda_i\,\mathrm{E}[Y_i]}\right).$$

For *preemptive LIFO* queues, (8) similarly gives

$$\omega_{\mathbf{z}_i}^*(y) = \big(\sum_j \nu_j\big)\frac{1 + (y/\nu_i)\sum_{j=1}^{n_i} 1/\Delta_{i,j}^*}{\nu_i - \lambda_i\,\mathrm{E}[Y_i]},$$

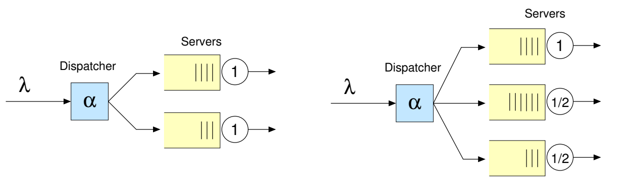
where n_i denotes the number of jobs in queue i and $\Delta_{i,j}^*$ the initial service requirement (in time) of job j in queue i. According to the FPI principle, one simply chooses the queue with the smallest (expected) cost,

$$\alpha_{\tilde{\mathbf{z}}}(y) \triangleq \underset{i}{\operatorname{argmin}}\, \omega_{\mathbf{z}_i}^*(y). \tag{12}$$

We refer to these improved dispatching policies simply as the *FPI-p policy*, where p denotes the basic dispatching policy, e.g., RND-opt or RND-ρ, where for brevity reasons the RND prefix is often omitted. Note that the factor $\sum_j \nu_j$ in both expressions is a common constant for all queues.

RND-ρ	state-independent random policy with load balancing
RND-opt	state-independent random policy minimizing the mean slowdown (assumes LIFO/PS)
SITA-E	state-independent size-interval-task-assignment with equal loads (optionally, with switch)
Round-Robin	assigns arriving tasks sequentially to servers [8]
LWL$^-$	least-work-left, assigns a job to server with the least amount unfinished work upon arrival
LWL$^+$	same as LWL$^-$ but based on the unfinished work (in time) including the new job [19]
JSQ	join-the-shortest-queue, i.e., the one with the least number of jobs [29]
Myopic	minimize the mean slowdown on condition that no further jobs arrive
FPI	first policy iteration on the state-independent RND-ρ, RND-opt and SITA-E policies

TABLE I
DISPATCHING POLICIES EVALUATED IN THE NUMERICAL EXAMPLES.



(a) Two identical servers (b) Three heterogeneous servers

Fig. 5. Example dispatching systems.

In a symmetric case of m identical servers and the RND-U basic policy, a corresponding FPI-based policy reduces to a well-known dispatching policy, i.e., with FIFO queues, FPI yields LWL, and with LIFO, the Myopic policy (see Table I). Moreover, given additionally a constant job size, then in case of LIFO queues one ends up with the JSQ policy. In general, for constant job sizes the slowdown objective reduces to the minimization of the mean sojourn time.

V. NUMERICAL EXAMPLES

Let us next evaluate by means of numerical simulations the improved dispatching policies derived in Section IV. To this end, we have chosen to consider two elementary server systems that are illustrated in Fig 5:

(a) Two identical servers with rates $\nu_1 = \nu_2 = 1$,
(b) Three heterogeneous servers with rates $\nu_1 = 1$ and $\nu_2 = \nu_3 = 1/2$.

Thus, the total service rate in both systems is equal to 2. We compare the mean slowdown performance of the FPI policies against several well-known heuristic dispatching policies, including the state-independent policies RND-opt, RND-ρ and SITA-E. The somewhat more sophisticated least-work-left (LWL) policies choose the queue with the least amount of unfinished work (backlog). The difference between the LWL policies is that LWL$^-$ considers the situation without the new job, and LWL$^+$ afterwards. With identical servers the LWL policies are equivalent. JSQ chooses the queue with the least number of jobs. With the LWL and JSQ, the ties are broken in favor of a faster server. The Myopic policy assumes in a greedy fashion that no further jobs arrive and chooses the queue which minimizes the immediate cost the known jobs accrue. All policies are listed in Table I.

In many real-life settings, the job sizes have been found to exhibit a heavy-tailed behavior, (cf., e.g., file sizes in the Internet). Thus, in most examples we assume a *bounded Pareto distribution* with pdf

$$f(x) = \frac{\alpha k^\alpha}{1-(k/p)^\alpha}\, x^{-\alpha-1}, \quad k \le x \le p, \qquad (13)$$

where $(k, p, \alpha) = (0.33959, 1000, 1.5)$ so that $\mathrm{E}[X] \approx 1$. This job size distribution is particularly suitable for SITA-E.

A. FIFO

First we assume that servers operate under the FIFO scheduling discipline and job sizes obey the bounded Pareto distribution (13). With FIFO, the departure time gets fixed at dispatching and the Myopic policy corresponds to selfish users choosing the queue that guarantees the shortest sojourn time, i.e., LWL$^+$. Similarly, LWL$^-$ is equivalent to an M/G/m system with a single shared queue.

Two identical servers: Fig. 6 (left) depicts the results with two identical servers with rates $\nu_1 = \nu_2 = 1$. The x-axis corresponds to the offered load ρ and the y-axis to the relative mean slowdown, $\mathrm{E}[\gamma]/\mathrm{E}[\gamma_{\text{SITA-E}}]$, i.e., the comparison is against SITA-E. We find that the policies appear to fall in one of the three groups with respect to slowdown: SITA policies form the best group, other queue state-dependent policies the next, and RND-U and Round-Robin have the worst performance. Especially, FPI-SITA-E outperforms the other policies by a significant margin, including the other SITA policies. Note also that when ρ is small, the sensible state-dependent policies utilize idle servers better than, e.g., SITA-E.

Fig. 7 illustrates SITA-E and FPI-SITA-E in the same setting at an offered load of $\rho = 0.5$. The x- and y-axes correspond to the backlog in Queue 1 and Queue 2, respectively, and the z-axis to the maximum job size a given policy assigns to Queue 1. With SITA-E, this threshold is constant, while FPI-SITA-E changes the threshold dynamically. One observes that as a result of FPI, Queue 1 has become a “high-priority” queue where no jobs are accepted whenever Queue 1 has more unfinished work than Queue 2. Similarly, when Queue 2 has a long backlog, the threshold for assigning a job to Queue 1 becomes higher with FPI than with SITA-E.

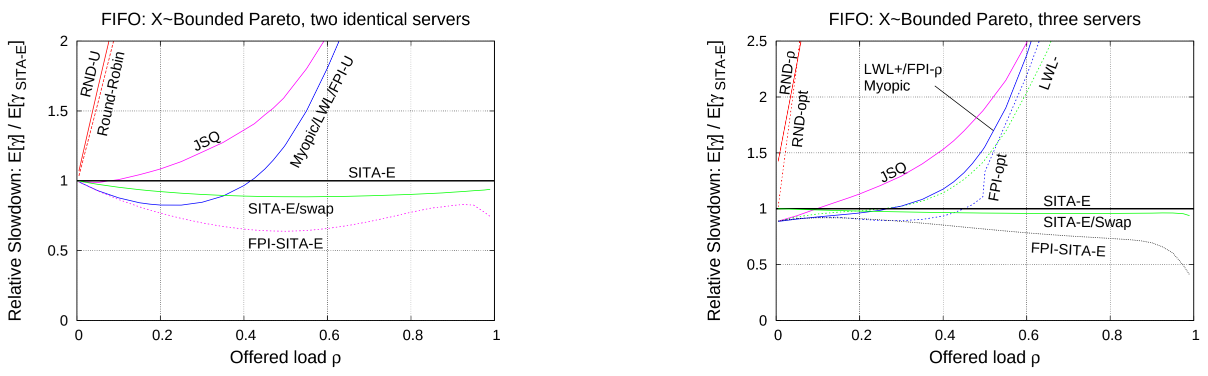


Fig. 6. Numerical results with two identical (left) and three heterogeneous servers having rates $\boldsymbol{\nu} = (\mathbf{1}, \mathbf{0.5}, \mathbf{0.5})$ (right). The scheduling discipline is FIFO and jobs sizes obey bounded Pareto distribution.

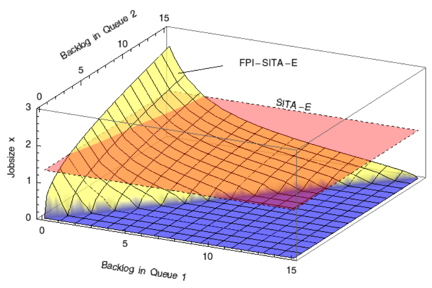


Fig. 7. SITA-E and FPI-SITA-E policies in the setting of two identical FIFO queues, bounded Pareto distributed job sizes and offered load $\boldsymbol{\rho} = \mathbf{0.5}$. $\mathbf{x}$- and $\mathbf{y}$-axes correspond to the backlog in Queue 1 and Queue 2, respectively, and $\mathbf{z}$-axis to the maximum job size that a policy assigns to Queue 1.

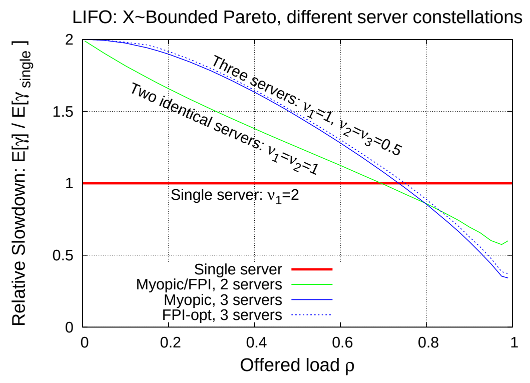


Fig. 9. Comparison of the chosen server constellations with an equal total capacity: i) single server, ii) two identical servers, and iii) three servers.

Three heterogeneous servers: Consider next the asymmetric setting comprising one primary server with service rate $\nu_1 = 1$ and two secondary servers with rates $\nu_2 = \nu_3 = 1/2$. Fig. 6 (right) illustrates the results from a numerical simulation.

The state-independent random policies are much worse than the other policies. However, state-independent SITA-E again proves out to be better than the state-dependent policies JSQ, LWL$^-$, LWL$^+$, Myopic, FPI-ρ and FPI-opt. The gain from switching the roles of the queues (SITA-E vs. SITA-Es) is smaller in this case due to the fact that we may only switch the roles of the two slower queues having the same service rate $\nu = 0.5$. As expected, the FPI-SITA-E policy yields a significantly lower mean slowdown than any other policy especially under a heavy load.

Based on the results, one can assume that the relative values obtained for the random policies simply do not capture the situation sufficiently well, while SITA-E is already a good dispatching policy, and FPI then leads to "adaptive" size intervals.

B. Preemptive LIFO

Next we assume that the servers are bound to operate under LIFO, which, as mentioned, has a reasonably robust performance with respect to the mean slowdown. Job sizes are again assumed to obey the bounded Pareto distribution.

Two identical servers: Fig. 8 (left) illustrates the performance with respect to the slowdown criterion for two identical servers. On x-axis is again the offered load ρ and y-axis corresponds to the relative mean slowdown (here comparison is against the Myopic policy). One can identify three performance groups: RND and LWL form the worst performing group, then come Round-Robin and JSQ, and the Myopic and FPI-RND policy achieve the lowest mean slowdown.

Three heterogeneous servers: Fig. 8 (right) illustrates the simulation results in the asymmetric setting comprising one primary server with service rate $\nu_1 = 1$ and two secondary servers with rates $\nu_2 = \nu_3 = 1/2$. In this case, the Myopic approach is the optimal (among the candidates) while both FPI policies attain almost identical mean slowdown. Even though the value function for the Myopic policy is not available for us, one can estimate the relative values by means of simulation and carry out the policy improvement numerically. However, in this paper we do not pursue into this direction.

Server constellations: In Fig. 9 we compare the mean slowdown between different server constellations assuming the preemptive LIFO scheduling discipline. The y-axis represents the relative performance when compared to the two server system. To no surprise, when the load is small or moderate, a single server system achieves the lowest mean slowdown.

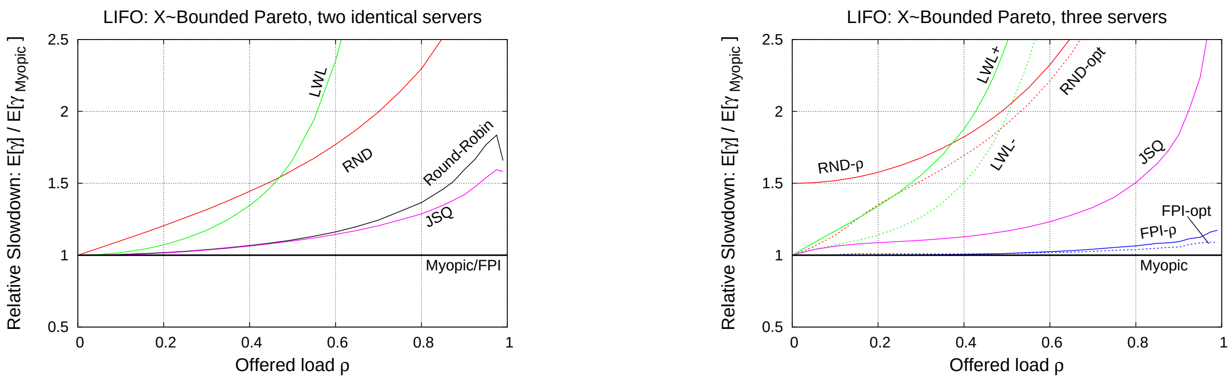


Fig. 8. Numerical results with preemptive LIFO scheduling discipline for two identical (left) servers with rates $\nu_1 = \nu_2 = 1$, and three heterogeneous servers with rates $\nu_1 = 1$ and $\nu_2 = \nu_3 = 1/2$.

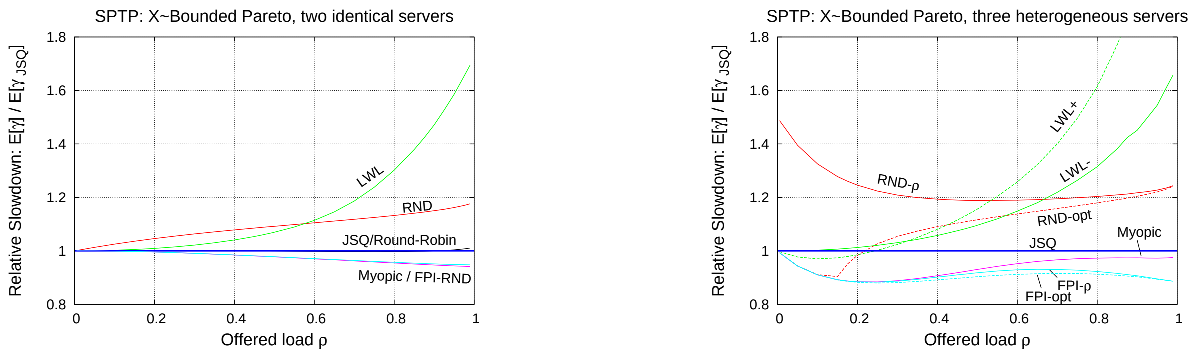


Fig. 10. Numerical results with SPTP for two identical (left) and three heterogeneous servers (right).

However, as the load increases more, first the two server system becomes optimal, and then eventually the three server system takes the lead. This is due to the fact that at higher load levels, the correct dispatching decisions allow one to serve more expensive jobs faster. One interesting research question indeed is that given a total capacity budget, what is the optimal server constellation so as to minimize the mean slowdown.

C. SPTP

Finally, let us consider the SPTP scheduling discipline that is the natural choice when minimizing the mean slowdown (see Section II). The job sizes are again assumed to obey the bounded Pareto distribution. Fig. 10 (left) illustrates the slowdown performance with the two identical servers. Interestingly, LWL becomes even weaker than RND-U as the offered load increases. Myopic and FPI-RND achieve the lowest mean slowdown.

In the case of three heterogeneous servers, the situation is again more challenging and the numerical results are depicted in Fig. 10 (right). At low levels of load, RND-opt is a surprisingly good choice, suggesting that SPTP manages to locally "correct" the occasional suboptimal decisions. When the load increases, the LWL policies become weak again. The Myopic policy shows a robust and good performance at all levels of offered load. However, the FPI policies, and FPI-opt in particular, achieve the lowest mean slowdown which is significantly better than with any other policy when $\rho > 0.5$.

D. Comparison of scheduling disciplines

So far we have fixed a scheduling discipline and evaluated different dispatching policies. Here we give a brief comparison of different scheduling disciplines with respect to the slowdown metric. Job sizes obey either (a) uniform distribution, $Y \sim \mathrm{U}(0.5, 1.5)$, or (b) the bounded Pareto distribution, both having the same mean, $\mathrm{E}[Y] = 1$.

First we consider a *single server queue*. Fig. 11 (left) depicts a comparison against LIFO with the same job size distribution on the logarithmic scale. In accordance with (3), FIFO is better than LIFO with uniform job size distribution, and with the bounded Pareto distribution the situation is the opposite. SPTP is clearly the best in both cases. In Fig. 11 (right), the reference level is the mean slowdown with SPTP. Here we have included also SRPT, which turns out to be only marginally better than SRPT, as observed also in [28]. LIFO and PS are significantly worse.

Fig. 12 illustrates the relative performance between different scheduling disciplines and dispatching policies in the heterogeneous three server dispatching system. SRPT is not included as its performance is very similar to SPTP. The reference level is the mean slowdown a JSQ/FIFO (left) and JSQ/LIFO (right) achieve. FPI manages to outperform the corresponding JSQ policy in all cases. Especially with the bounded Pareto

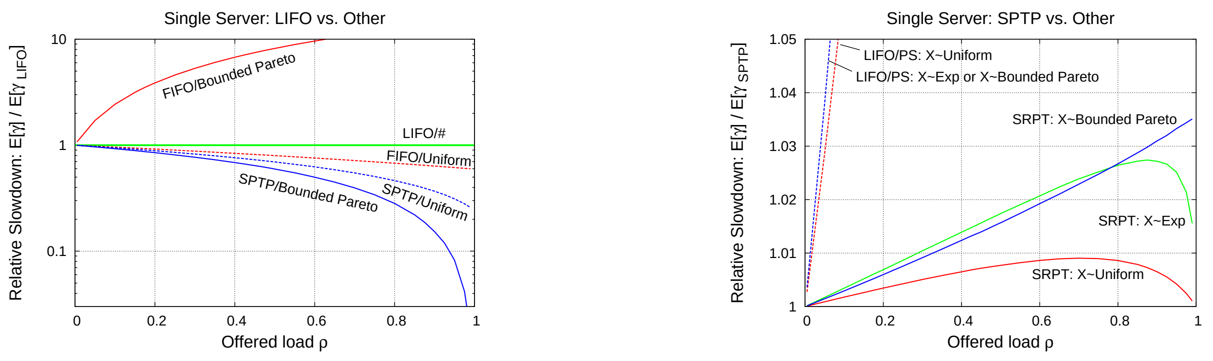
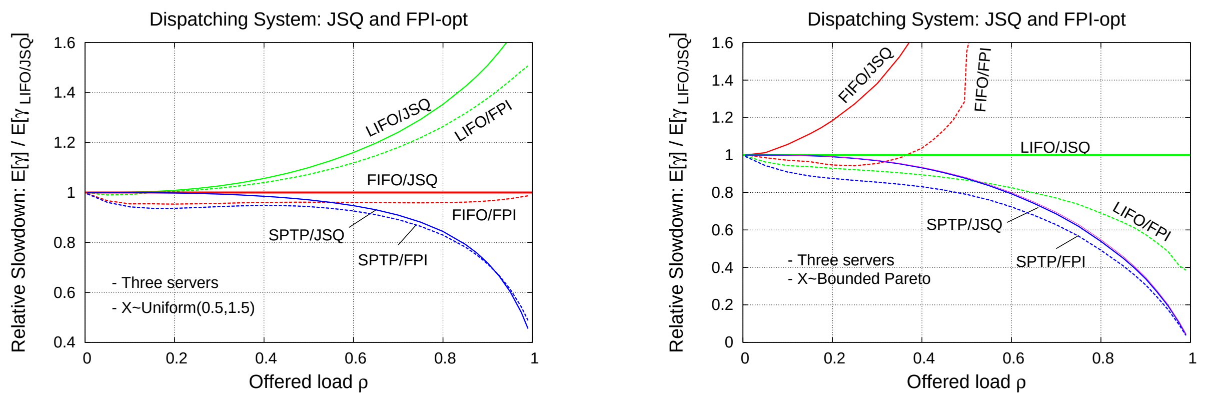


Fig. 11. Comparison of different scheduling disciplines in a single server queue. FIFO is better than LIFO with some job size distributions (left). SPTP is only marginally better than SRPT (right).



(a) $X \sim \mathrm{U}(0.5, 1.5)$ (b) $X \sim$ Bounded Pareto

Fig. 12. Heterogeneous dispatching system with different scheduling disciplines.

distributed job sizes, FPI/LIFO performs rather well when compared to JSQ/LIFO. SPTP is clearly superior scheduling discipline in both cases, as expected.

In general, the scheduling discipline seems to have a stronger influence to the performance than the dispatching policy.

VI. Conclusions

This work generalizes the earlier results of the size-aware value functions with respect to the sojourn time for M/G/1 queues with FIFO, LIFO, SPT and SRPT disciplines to a general job specific holding cost. Additionally, we have considered the SPTP queueing discipline, the optimality of which with Poisson arrivals was also established. As an important special case, one obtains the *slowdown criterion*, where the holding cost is inversely proportional to the job's (original) size. These results were then utilized in developing robust policies for dispatching systems. In particular, the value functions enable the policy improvement step for an arbitrary state-independent dispatching policy such as Bernoulli-splitting. The derived dispatching policies were also shown to perform well by means of numerical simulations. In particular, the highest improvements were often obtained in a more challenging heterogeneous setting with unequal service rates. Also the SPTP scheduling discipline outperformed the other by a clear margin in the examples, except the SRPT discipline, which appears to offer a rather similar performance in terms of slowdown and sojourn time.

References

[1] S. Aalto, U. Ayesta, S. Borst, V. Misra, and R. Núñez-Queija. Beyond processor sharing. *SIGMETRICS Perform. Eval. Rev.*, 34:36–43, 2007.

[2] S. Aalto, U. Ayesta, and R. Righter. On the Gittins index in the M/G/1 queue. *Queueing Systems*, 63(1-4):437–458, 2009.

[3] S. Aalto and J. Virtamo. Basic packet routing problem. In *NTS-13*, Trondheim, Norway, Aug. 1996.

[4] E. Bachmat and H. Sarfati. Analysis of SITA policies. *Performance Evaluation*, 67(2):102–120, 2010.

[5] C. E. Bell and J. Shaler Stidham. Individual versus social optimization in the allocation of customers to alternative servers. *Management Science*, 29(7):831–839, July 1983.

[6] M. E. Crovella, M. Harchol-Balter, and C. D. Murta. Task assignment in a distributed system: Improving performance by unbalancing load. In *ACM SIGMETRICS*, pages 268–269, June 1998.

[7] A. Downey. A parallel workload model and its implications for processor allocation. In *IEEE HPDC'97*, pages 112–123, Aug. 1997.

[8] A. Ephremides, P. Varaiya, and J. Walrand. A simple dynamic routing problem. *IEEE Transactions on Automatic Control*, 25(4):690–693, Aug. 1980.

[9] H. Feng and V. Misra. Mixed scheduling disciplines for network flows. *SIGMETRICS Perform. Eval. Rev.*, 31:36–39, Sept. 2003.

[10] H. Feng, V. Misra, and D. Rubenstein. Optimal state-free, size-aware dispatching for heterogeneous M/G/-type systems. *Perform. Eval.*, 62(1-4), 2005.

[11] J. Gittins. *Multi-armed Bandit Allocation Indices*. Wiley, 1989.

[12] V. Gupta, M. Harchol-Balter, K. Sigman, and W. Whitt. Analysis of join-the-shortest-queue routing for web server farms. *Performance Evaluation*, 64(9-12):1062–1081, Oct. 2007.
[13] M. Harchol-Balter. Task assignment with unknown duration. *J. ACM*, 49(2):260–288, Mar. 2002.
[14] M. Harchol-Balter, M. E. Crovella, and C. D. Murta. On choosing a task assignment policy for a distributed server system. *Journal of Parallel and Distributed Computing*, 59:204–228, 1999.
[15] M. Harchol-Balter and A. B. Downey. Exploiting process lifetime distributions for dynamic load balancing. *ACM Transactions on Computer Systems*, 15(3):253–285, Aug. 1997.
[16] M. Harchol-Balter, A. Scheller-Wolf, and A. R. Young. Surprising results on task assignment in server farms with high-variability workloads. In *ACM SIGMETRICS*, pages 287–298, 2009.
[17] M. Harchol-Balter, K. Sigman, and A. Wierman. Asymptotic convergence of scheduling policies with respect to slowdown. *Perform. Eval.*, 49(1-4):241–256, Sept. 2002.
[18] M. Haviv and T. Roughgarden. The price of anarchy in an exponential multi-server. *Oper. Res. Lett.*, 35(4):421–426, 2007.
[19] E. Hyytiä, A. Penttinen, and S. Aalto. Size- and state-aware dispatching problem with queue-specific job sizes. *European Journal of Operational Research*, 217(2):357–370, Mar. 2012.
[20] E. Hyytiä, A. Penttinen, S. Aalto, and J. Virtamo. Dispatching problem with fixed size jobs and processor sharing discipline. In *ITC'23*, SFO, USA, Sept. 2011.
[21] E. Hyytiä, J. Virtamo, S. Aalto, and A. Penttinen. M/M/1-PS queue and size-aware task assignment. *Performance Evaluation*, 68(11):1136–1148, Nov. 2011.
[22] K. R. Krishnan. Joining the right queue: a state-dependent decision rule. *IEEE Transactions on Automatic Control*, 35(1):104–108, Jan. 1990.
[23] Z. Liu and R. Righter. Optimal load balancing on distributed homogeneous unreliable processors. *Operations Research*, 46(4):563–573, 1998.
[24] Z. Liu and D. Towsley. Optimality of the round-robin routing policy. *Journal of Applied Probability*, 31(2):466–475, June 1994.
[25] L. Schrage. A proof of the optimality of the shortest remaining processing time discipline. *Operations Research*, 16(3), 1968.
[26] W. Whitt. Deciding which queue to join: Some counterexamples. *Oper. Res.*, 34(1):55–62, 1986.
[27] A. Wierman. Fairness and scheduling in single server queues. *Surveys in Operations Research and Management Science*, 16(1):39–48, 2011.
[28] A. Wierman, M. Harchol-Balter, and T. Osogami. Nearly insensitive bounds on SMART scheduling. In *ACM SIGMETRICS*, pages 205–216, 2005.
[29] W. Winston. Optimality of the shortest line discipline. *Journal of Applied Probability*, 14:181–189, 1977.
[30] S. Yang and G. de Veciana. Size-based adaptive bandwidth allocation: optimizing the average QoS for elastic flows. In *IEEE INFOCOM*, 2002.

Appendix

Carrying out the similar steps as with SPTP, it is straightforward to derive value function also in the case of SPT and SRPT policies. Again, we let $f(x)$ denote the pdf of the job size distribution and $\rho(x)$ the offered load due to jobs shorter than x according to (9). The backlog due to higher priority work in this case is $u_{\mathbf{z}}(x)$, i.e., where the priority index for SPT is the initial service time Δ_i^*, and for SRPT the remaining service time Δ_i.

A. M/G/1-SPT

We consider the non-preemptive SPT, which is the optimal non-preemptive schedule with respect to both the mean sojourn time and the mean slowdown. Thus, the remaining service time of a queueing job is equal to the initial service time. Therefore, a sufficient state description for a non-preemptive M/G/1-SPT queue with arbitrary holding costs is $\mathbf{z} = ((\Delta_1, a_1); \ldots ; (\Delta_n, a_n))$, where job 1 (if any) is currently receiving service and jobs $2, \ldots, n$ are waiting in the queue. Without lack of generality, we can assume that $\Delta_2 < \Delta_3 < \ldots < \Delta_n$.

Proposition 4: For the size-aware value function with respect to arbitrary holding costs in an non-preemptive M/G/1-SPT queue it holds that,

$$v_{\mathbf{z}} - v_0 = \sum_{i=1}^{n} b_i \left(\Delta_i + \frac{\sum_{j=1}^{i-1} \Delta_j}{1-\rho(\Delta_i)} \right) + \frac{\lambda}{2} \sum_{i=1}^{n} \left(\left(\left(\sum_{j=1}^{i} \Delta_j \right)^2 + \sum_{j=i+1}^{n} (\Delta_j)^2 \right) \int_{\tilde{\Delta}_i}^{\tilde{\Delta}_{i+1}} \frac{b(x)\, f(x)}{(1-\rho(x))^2}\, dx \right), \tag{14}$$

where $b(x) = \mathrm{E}[B \mid X{=}x]$ and the integration intervals are

$$\tilde{\Delta}_i = \begin{cases} 0, & i = 1 \\ \Delta_i, & i = 2, \ldots, n \\ \infty, & i = n+1. \end{cases}$$

Proof: Similarly as with the SPTP, we compare two systems: System 1 initially in state $\mathbf{z}$ and System 2 initially empty. The two systems behave identically once System 1 becomes empty for the first time. We can write $v_{\mathbf{z}} - v_0 = h_1 + h_2$, where h_1 is the cost the n jobs initially present (only) in System 1 incur, and h_2 the difference in cost the later arriving customers incur between the two systems.

First, the remaining sojourn time of job i in a non-preemptive M/G/1-SPT queue is

$$\mathrm{E}[R_i] = \Delta_i + \frac{\sum_{j=1}^{i-1} \Delta_j}{1-\rho(\Delta_i)}, \tag{15}$$

which holds for all n jobs present only in System 1. Therefore, their contribution to the $v_{\mathbf{z}} - v_0$ is

$$h_1 = \sum_{i=1}^{n} b_i \left(\Delta_i + \frac{\sum_{j=1}^{i-1} \Delta_j}{1-\rho(\Delta_i)} \right).$$

For the later arriving jobs, we condition the derivation on job sizes $(x, x{+}dx)$, which arrive at rate of $\lambda\, f(x)\, dx$. Let $\tilde{t}(x)$ denote the mean additional sojourn time such jobs experience in System 1 when compared to System 2. Let $b(x) = \mathrm{E}[B \mid X{=}x]$ so that we have an expression for h_2

$$h_2 = \int_0^\infty \tilde{t}(x) \cdot b(x)\, dx.$$

Instead of considering actual arrivals, we focus on a rate at which *virtual costs* are accrued in order to find $\tilde{t}(x)$. With aid of (15), one can deduce that the virtual cost rate is equal to

$$\lambda f(x)\, \frac{U_{\mathbf{z}}(x,t)}{1-\rho(x)},$$

where $U_{\mathbf{z}}(x,t)$ denotes the amount of work at time t that would be processed before an arriving size x job when a system was initially in state $\mathbf{z}$. In particular, we can write

$$\tilde{t}(x) = \mathrm{E}[\int_0^\infty \lambda\, f(x) \left(\frac{U_{\mathbf{z}}(x,t)}{1-\rho(x)} - \frac{U_0(x,t)}{1-\rho(x)} \right) dt],$$

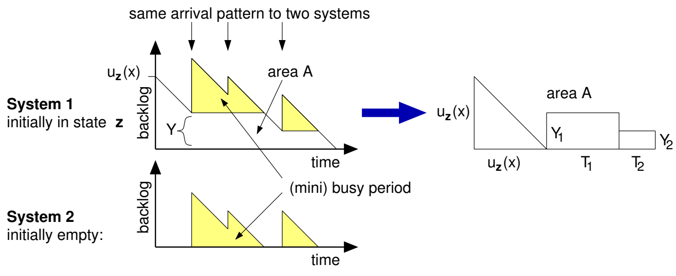


Fig. 13. Derivation of value function for an M/G/1-SPT queue.

which gives

$$\tilde{t}(x) = \frac{\lambda f(x)}{1-\rho(x)} \mathrm{E}[\int_0^\infty U_{\mathbf{z}}(x,t) - U_{\mathbf{z}}(x,t)\,dt].$$

Let $k_{\mathbf{z}}(x)$ denote the number of jobs waiting in the queue with a remaining service time greater than x.

$$k_{\mathbf{z}}(x) = |\{i \in \{2,\dots,n\} \,:\, \Delta_i > x\}|,$$

Due to the non-preemptive discipline, the evolution of $U_{\mathbf{z}}(x,t)$ for $\mathbf{z} \neq 0$ consists of $k_{\mathbf{z}}(x)+1$ phases. Let $m = n - k_{\mathbf{z}}(x)$. During the first phase jobs $1,\dots,m$ are served and the initial backlog size x jobs see is $\Delta_1 + \dots + \Delta_m$. The second phase starts when job $m+1$ enters the server and cannot be preempted, thus creating an initial backlog Δ_{m+1} for size x jobs, etc.

Considering arrival sample paths, one can deduce that the difference $\int_0^\infty U_{\mathbf{z}}(x,t) - U_{\mathbf{z}}(x,t)\,dt$ corresponds to the white marked region in Fig. 13, which consists of $k_{\mathbf{z}}(x)+1$ statistically independent areas. The area of a phase starting with an initial backlog of u from size x customers' point of view is given by

$$\frac{u^2}{2} + \lambda F(x)\, u \cdot \frac{\mathrm{E}[X \mid X < x]}{1-\rho} \cdot \frac{u}{2} = \frac{u^2}{2(1-\rho(x))}. \qquad (16)$$

Defining

$$g(x) = \frac{b(x)\, f(x)}{(1-\rho(x))^2},$$

then gives

$$\begin{aligned} h_2 &= \frac{\lambda}{2}(\Delta_1^2 + \dots + \Delta_n^2) \int_0^{\Delta_2} g(x)\,dx \\ &+ \frac{\lambda}{2}((\Delta_1+\Delta_2)^2 + \Delta_3^2 + \dots + \Delta_n^2) \int_{\Delta_2}^{\Delta_3} g(x)\,dx + \dots \\ &+ \frac{\lambda}{2}(\Delta_1 + \dots + \Delta_n)^2 \int_{\Delta_n}^{\infty} g(x)\,dx. \end{aligned}$$

For example, when $x > \Delta_n$, all current n jobs in System 1 have a higher priority constituting a single phase with $u = \Delta_1 + \dots + \Delta_n$, etc. Next define the integration intervals, $\tilde{\Delta}_1 = 0$, $\tilde{\Delta}_i = \Delta_i$ for $i = 2,\dots,n$, and $\tilde{\Delta}_{n+1} = \infty$. Substituting these into the above gives

$$h_2 = \frac{\lambda}{2} \sum_{i=1}^{n} \left((\sum_{j=1}^{i} \Delta_j)^2 + \sum_{j=i+1}^{n} \Delta_j^2)^2 \right) \int_{\tilde{\Delta}_i}^{\tilde{\Delta}_{i+1}} \frac{b(x)\, f(x)}{(1-\rho(x))^2}\,dx,$$

which completes the proof. ■

B. M/G/1-SRPT

While SPT was optimal in the class of non-preemptive schedules, the SRPT discipline is the optimal preemptive schedule [25]. For SRPT, a sufficient state description is $\mathbf{z} = ((\Delta_1, a_1); \dots; (\Delta_n, a_n))$, where, without loss of generality, we again can assume that job 1 is currently receiving service (if any) and $\Delta_1 < \Delta_2 < \dots < \Delta_n$. For the total higher priority workload we have $u_{\mathbf{z}}(\Delta_i) = \sum_{j=1}^{i-1} \Delta_j$.

Proposition 5: For the size-aware value function in an M/G/1-SRPT queue with arbitrary holding cost it holds that

$$\begin{aligned} v_{\mathbf{z}} - v_0 = \sum_{i=1}^{n} b_i &\left(\frac{\sum_{j=1}^{i-1} \Delta_j}{1-\rho(\Delta_i)} + \int_0^{\Delta_i} \frac{1}{1-\rho(t)}\,dt \right) \\ + \frac{\lambda}{2} \sum_{i=0}^{n} &\Big[(n-i) \int_{\tilde{\Delta}_i}^{\tilde{\Delta}_{i+1}} \frac{x^2\, b(x)\, f(x)}{(1-\rho(x))^2}\,dx \\ &+ \left(\sum_{j=1}^{i-1} \Delta_j \right) \int_{\tilde{\Delta}_i}^{\tilde{\Delta}_{i+1}} \frac{b(x)\, f(x)}{(1-\rho(x))^2}\,dx \Big] \end{aligned} \qquad (17)$$

where $b(x) = \mathrm{E}[B \mid X = x]$ and

$$\tilde{\Delta}_i = \begin{cases} 0, & i = 0 \\ \Delta_i, & i = 1,\dots,n \\ \infty, & i = n+1. \end{cases}$$

Proof: Again, we consider System 1 initially in state $\mathbf{z}$ and System 2 initially empty, and find expressions for h_1 and h_2 corresponding to the cost the n jobs initially present (only) in System 1 incur, and the difference in cost the later arriving customers incur between the two systems.

In a M/G/1-SRPT queue at state $\mathbf{z} = (\Delta_1,\dots,\Delta_n)$, the mean remaining sojourn time of a job with a (remaining) size Δ and an unfinished work $u_{\mathbf{z}}(\Delta)$ ahead in the queue is given by [19]

$$\mathrm{E}[R_{\mathbf{z}}(\Delta)] = \frac{u_{\mathbf{z}}(\Delta)}{1-\rho(\Delta)} + \int_0^{\Delta} \frac{1}{1-\rho(t)}\,dt, \qquad (18)$$

which gives the expected cost the current n jobs[4] present in System 1 incur,

$$h_1 = \sum_{i=1}^{n} b_i \left(\frac{\sum_{j=1}^{i-1} \Delta_j}{1-\rho(\Delta_i)} + \int_0^{\Delta_i} \frac{1}{1-\rho(t)}\,dt \right).$$

For the later arriving jobs, we can again write

$$\begin{aligned} \tilde{t}(x) &= \mathrm{E}\Big[\int_0^\infty \lambda f(x) \Big[\big(x + \frac{U_{\mathbf{z}}(x,t)}{1-\rho(x)} + \int_0^x \frac{\rho(t)}{1-\rho(t)}\,dt \big) \\ &\qquad - \left(x + \frac{U_0(x,t)}{1-\rho(x)} + \int_0^x \frac{\rho(t)}{1-\rho(t)}\,dt \right) \Big] \Big], \\ &= \frac{\lambda f(x)}{1-\rho(x)} \mathrm{E}\Big[\int_0^\infty U_{\mathbf{z}}(x,t) - U_0(x,t)\,dt \Big], \end{aligned}$$

which describes the expected difference in the cumulative sojourn times between System 1 and System 2 for size x jobs.

With SRPT also the jobs in System 1 initially longer than x affect the result as eventually, one at a time, they decrease

[4]Note that we assume the opposite order than [19].

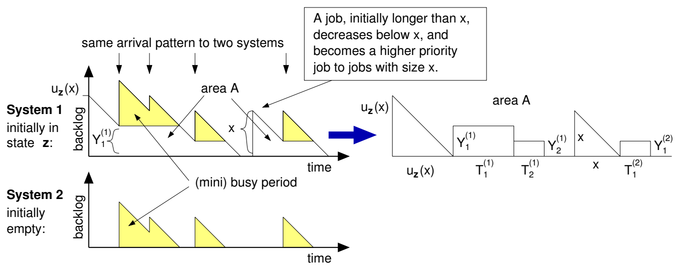


Fig. 14. Derivation of value function for an M/G/1-SRPT queue.

below the threshold x and trigger a new higher priority workload present only in System 1 (see Fig. 14). Consequently, we have one phase starting with a backlog of $u_{\mathbf{z}}(x)$, and then $n_{\mathbf{z}}(x)$ phases starting with a backlog of x, where $n_{\mathbf{z}}(x)$ denotes the number of jobs longer than x in state $\mathbf{z}$. In some sense, this is the same as with SPT, but the difference is the initial backlog size x jobs see: with SPT it is Δ_i instead of x for later phases. Hence, each of these phases reduce to (16), which gives

$$\tilde{t}(x) = \frac{\lambda\, f(x)\, \left[u_{\mathbf{z}}(x)^2 + n_{\mathbf{z}}(x)\, x^2\right]}{2(1-\rho(x))^2},$$

As $h_2 = \int b(x)\, \tilde{t}(x)\, dx$, and both $u_{\mathbf{z}}(x)$ and $n_{\mathbf{z}}(x)$ are (monotonic) step functions having jumps at $x = \Delta_1, \ldots, \Delta_n$, we proceed by integrating in parts. Defining

$$g(x) = \frac{f(x)\, b(x)}{(1-\rho(x))^2},$$

then gives

$$\begin{aligned} h_2 = {} & \frac{\lambda}{2} \int_0^{\Delta_1} n\, x^2\, g(x)\, dx \\ & + \frac{\lambda}{2} \int_{\Delta_1}^{\Delta_2} (\Delta_1^2 + (n-1)\, x^2)\, g(x)\, dx + \ldots \\ & + \frac{\lambda}{2} \int_{\Delta_n}^{\infty} (\Delta_1 + \ldots + \Delta_n)^2\, g(x)\, dx, \end{aligned}$$

and recalling that $v_{\mathbf{z}} - v_0 = h_1 + h_2$ then gives (17). ■